Point Cloud Collision Detection

EG 2004

Jan Klein
Uni Paderborn

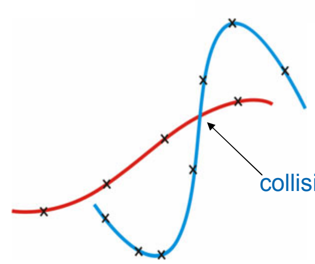
&

Gabriel Zachmann
Uni Bonn

Point Clouds

- Modern acquisition methods (scanning, sampling synthetic objects) lead to modern object representations.
- Efficient rendering (splatting & ray-tracing).
- Only very little work on interaction.


- Goal:
  - Fast collision detection between 2 point clouds.
  - No polygonal reconstruction.



Surface Definition

- Approximate surface by implicit function

$$S = \{x : f(x) = 0, x \in \mathbb{R}^3\}$$

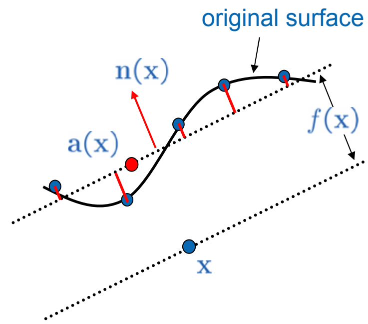
- Define $f(\mathbf{x})$ by weighted least squares:

1. $\mathbf{a}(\mathbf{x}) =$ weighted average of points.
2. $\mathbf{n}(\mathbf{x})$ using weighted least squares:

$$\sum_{i=1}^{N} (\mathbf{n}(\mathbf{x}) \cdot (\mathbf{a}(\mathbf{x}) - \mathbf{p}_i))^2 \theta(\|\mathbf{x} - \mathbf{p}_i\|)$$

Kernel: $\theta(d) = e^{-d^2/h^2}$

3. $f(\mathbf{x}) = \mathbf{n}(\mathbf{x}) \cdot (\mathbf{a}(\mathbf{x}) - \mathbf{x})$



Related Work

Geometric queries

- Approximating and Intersecting Surfaces from Points [Adamson & Alexa, 2003]

Boolean operations

- Shape Modeling with Point-Sampled Geometry [Pauly et al., 2003]
- Interactive Boolean Operations on Surfel Bounded Solids [Adams & Dutre, 2003]

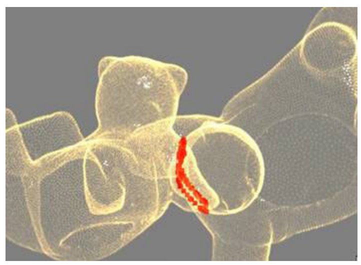
Time-critical algorithms

- Approximating Polyhedra with Spheres for Time-Critical Collision Detection [Hubbard, 1996]

Our Contribution

HEINZ NIXDORF INSTITUTE
University of Paderborn
Algorithms and Complexity

- Time-critical collision detection between point clouds.
- Point cloud hierarchy with low memory consumption.
- Traversal criterion allows for quick convergence.
- Randomized intersection tests in leaf nodes.



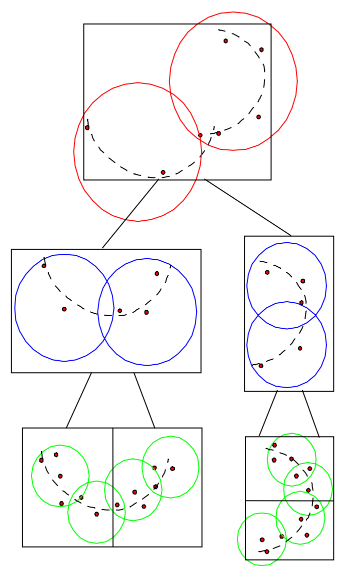
Overview of Point Cloud Hierarchy

HEINZ NIXDORF INSTITUTE
University of Paderborn
Algorithms and Complexity

Levels represent different resolutions of the surface (LODs).

1. - Points in leafs make up the whole point cloud.
   - Hierarchy according to volume criterion.
2. Subsampling and sphere covering at nodes.

→ Efficient storage



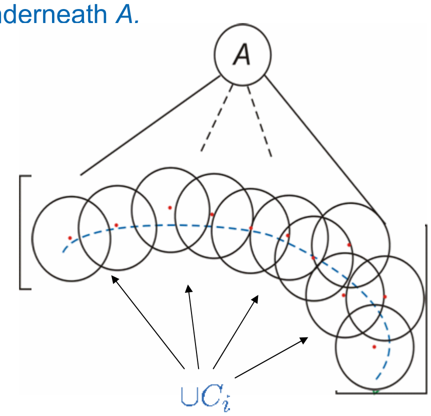
Requirements of Sphere Covering

Observation: surface is inside the set of convex hulls C_i of the leafs underneath *A*.

For each node *A*: find spheres K_j

- that cover $\cup C_i$
- with $Vol(\cup K_j)$ minimal
- same radius, number < c, centers $k_j \in P_A$

$\rightarrow c$ sample points and 1 radius per node.



Constructing the Sphere Covering

Construct sample in BV *A*:

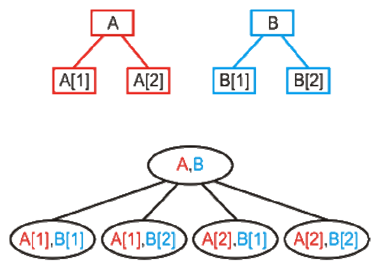
- choose sample points $\in A$ so that distances between them are of the same order.
- avoid points close to the border of $\cup C_i$.

Determine common radius r_A analogously to Monte-Carlo integration:

- repeat until Prob(spheres cover surface) is high enough:
  - generate randomly, independently test point p in $\cup C_i$.
  - if $p \notin \cup K_j$
    $r_A =$ minimal distance of p to a sample point.

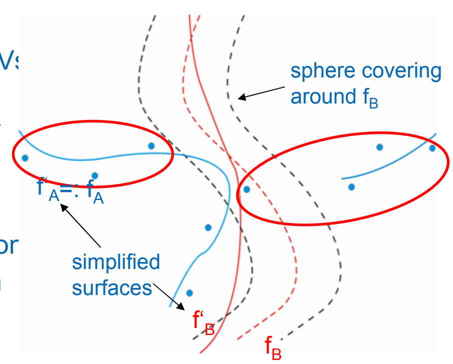
Overview of Collision Detection

- Simultaneous traversal, use BVs for overlap test.
- During traversal descend first into pairs with largest priority (use sample points).
- Leaf nodes:
  - estimate distance between surfaces
  - report a collision, if distance $< d_\epsilon$.



Traversal Criterion

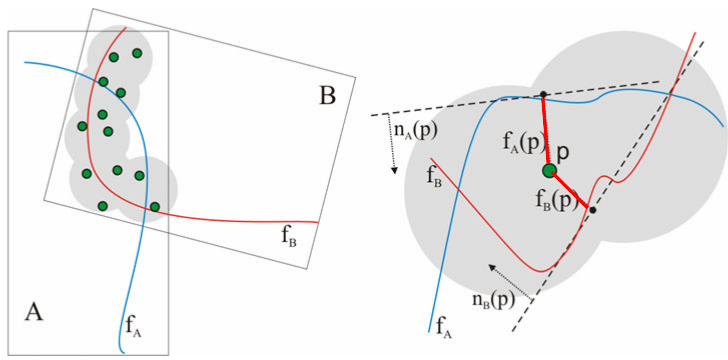
- If there are points on simplified surface of *A* that are on different sides of f'_B
  - then intersection is very likely.
  - give priority to those pairs of BVs
- Use the sign of *f'* as an indicator of the local "sideness".
- Estimate likelihood of an intersection proportional to number of points on both sides.



Collision Detection in Leaves

HEINZ NIXDORF INSTITUTE
University of Paderborn
Algorithms and Complexity

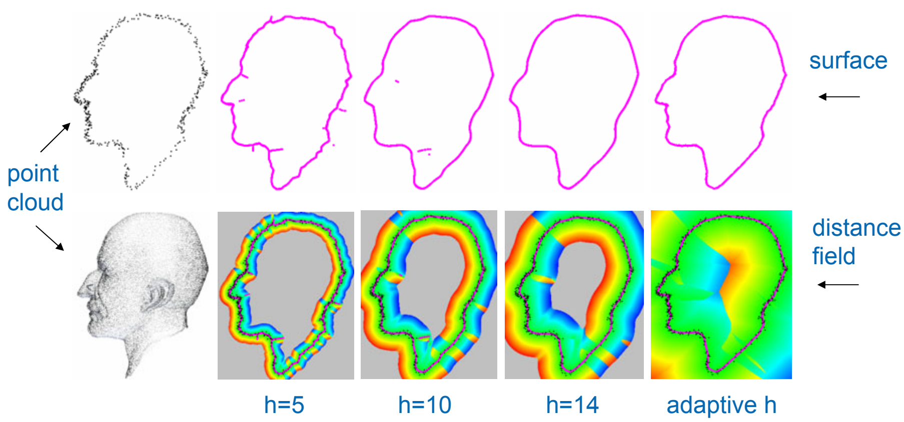
- Conceptually, find test point *p* with $f_A(\mathrm{p}) = f_B(\mathrm{p}) = 0$
  $\rightarrow$ too expensive.
- Generate randomly and independently a constant number of test points.
- $d_{AB} \approx \min_p\{|f_A(\mathrm{p})| + |f_B(\mathrm{p})|\}$ and report collision, if $d_{AB} < d_\epsilon$.



Automatic Bandwidth

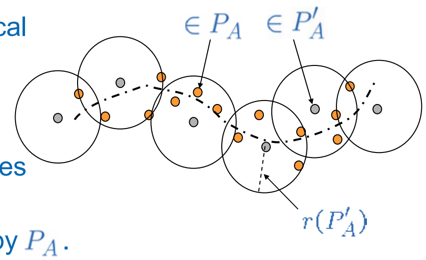
HEINZ NIXDORF INSTITUTE
University of Paderborn
Algorithms and Complexity

- BV hierarchy leads to sets with very different sampling densities.
- Which *h* in $\theta(d) = e^{-d^2/h^2}$?
  - *h* too small $\rightarrow$ variance
  - *h* too large $\rightarrow$ missing detail



Automatic Bandwidth Detection

HEINZ NIXDORF INSTITUTE
University of Paderborn
Algorithms and Complexity



- Bandwidth *h* should be adapted to local sampling density.

- Sample points $P'_A \subset P_A$,
  $r(P'_A)$: smallest radius, so that spheres centered at P'_A with that radius cover surface defined by P_A.

- Determine *h* from $r(P'_A)$: $h = \frac{m \cdot r(P'_A)}{\sqrt{|\log \theta_\epsilon|}}$

- $r(P'_A) = r_A$ or better: $r(P'_A) = \sqrt{|P_A|/|P'_A|} \cdot r(P_A)$.

- The number of sample points per node can be derived to achieve a certain sampling radius.

Time-Critical Collision Detection

HEINZ NIXDORF INSTITUTE
University of Paderborn
Algorithms and Complexity

- Two goals:
  - If time budget is exhausted, stop collision detection and return “best effort” answer.
  - If there is still time left, spend more time on the collision detection in leaves to increase the accuracy.

- Spend the same time *t* for each single collision query by adjusting
  - the number of test points and
  - the distance d_ϵ that has to be found between the objects.

Benchmark

HEINZ NIXDORF INSTITUTE
University of Paderborn
Algorithms and Complexity

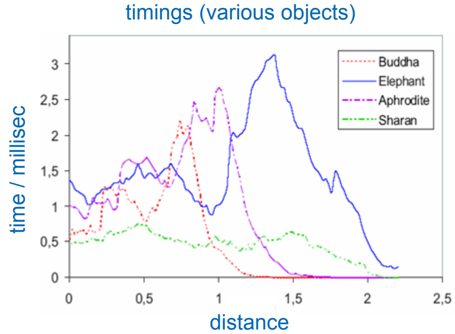
- For range of distances: average collision detection time for a complete revolution (5000 steps).
- Objects are scaled uniformly.

Pentium-IV, 2.8 GHz, 1 GB main memory.

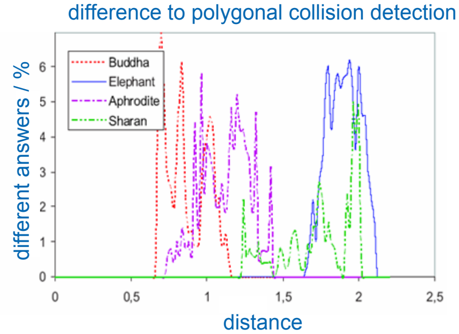
Time and Quality

HEINZ NIXDORF INSTITUTE
University of Paderborn
Algorithms and Complexity

timings (various objects)

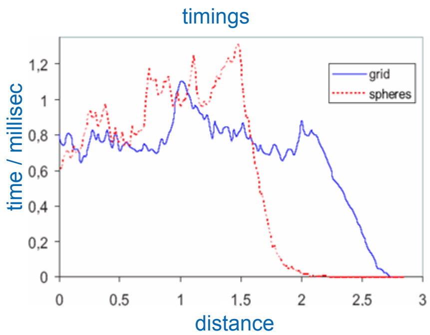


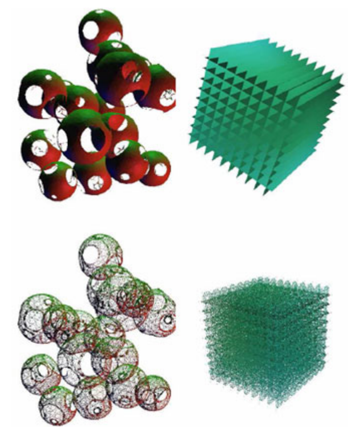
difference to polygonal collision detection



- Average runtime is between 0.5 and 3.0 millisec $\rightarrow$ real-time applications
- Differences can be explained by:
  - surface defined by vertices of polygonal object is different from polygonal model.
  - intersection finding algorithm in leaf nodes is still simplistic.

Artificial Models

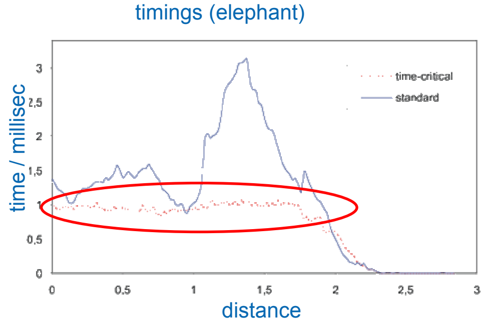


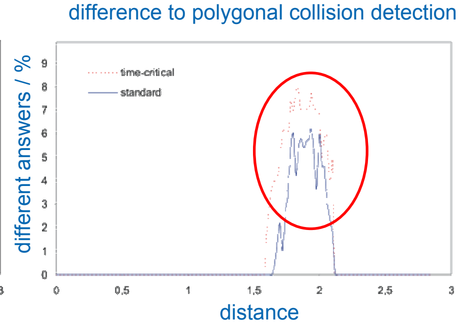


- Boundaries, several unconnected components.
- Timings as good as for other models.
- Grid model causes up to 10% differences.
- Surface definition is only for manifold objects.

≈ 135,000 points ≈ 200,000 points

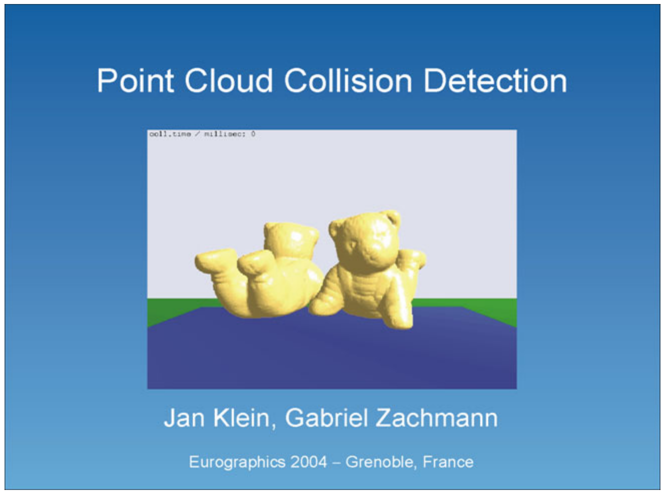
Time-Critical Collision Detection





- Timings are bounded by t = 1 millisec.
- Differences to polygonal collision detection slightly increase.

Video



Conclusion & Future Work

HEINZ NIXDORF INSTITUTE
University of Paderborn
Algorithms and Complexity

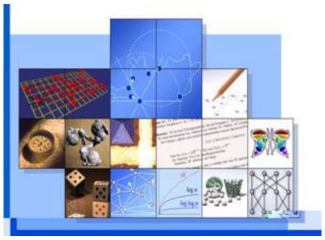
Conclusion

- Fast and time-critical collision detection of point clouds.
- Traversal criterion allows for guiding the traversal.
- Fast construction of hierarchical sphere covering of point cloud.
- Only small differences compared to polygonal collision detection.

Future Work

- Performance and accuracy can be increased:
  - faster convergence in leaves
  - point hierarchy and sphere coverings could be improved.
- Use surface definition based on proximity graphs [Siggraph 2004 sketch].

Thank you!



Jan Klein

Heinz Nixdorf Institute and
Institute of Computer Science
University of Paderborn, Germany

janklein@uni-paderborn.de

Dr. Gabriel Zachmann

Dept. of Computer Graphics and
Virtual Reality
University of Bonn, Germany

zach@cs.uni-bonn.de